

# Snackbar

Snackbar A flexible, accessible snackbar component for transient notifications, actions, and feedback. Supports multiple positions, variants, icons, actions, and theming. Use snackbars to provide brief messages about app processes, confirmations, or errors, with optional actions.

How to use ■ `import { SnackbarService } from "@aava/play-core"` ; Basic Usage ■ Show a simple snackbar with a message.

Positions ■ Display snackbars in different screen positions. There are 7 common positions used to describe snackbars placement.

- `top-left` : Positioned at the upper-left corner.
- `top-center` : Positioned at the top edge, horizontally centered.
- `top-right` : Positioned at the upper-right corner.
- `bottom-left` : Positioned at the lower-left corner.
- `bottom-center` : Positioned at the bottom edge, horizontally centered.
- `bottom-right` : Positioned at the lower-right corner.
- `center` : Positioned exactly in the middle, both vertically and horizontally centered.

Actions & Icons ■ Add action buttons, icons, make snackbars dismissible or persistent.

API Reference ■

Property	Type	Default	Description
<code>message</code>	<code>string</code>	<code>"</code>	Message text to display in the snackbar
<code>position</code>	<code>SnackbarPosition</code>	<code>'bottom-center'</code>	Position of the snackbar on screen
<code>duration</code>	<code>number</code>	<code>4000</code>	Duration in milliseconds before auto-dismiss
<code>color</code>	<code>string</code>	<code>"</code>	Custom text color for the snackbar
<code>backgroundColor</code>	<code>string</code>	<code>"</code>	Custom background color for the snackbar
<code>action</code>	<code>SnackbarAction</code>	<code>"</code>	Action button configuration
<code>icon</code>	<code>SnackbarIcon</code>	<code>"</code>	Icon configuration for the snackbar
<code>dismissible</code>	<code>boolean</code>	<code>false</code>	Whether the snackbar can be dismissed
<code>persistent</code>	<code>boolean</code>	<code>false</code>	Whether the snackbar persists until dismissed
<code>variant</code>	<code>'surface-bold'   string</code>	<code>'surface-bold'</code>	Visual variant of the snackbar
<code>type</code>	<code>'medium'   'strong'   'max'   string</code>	<code>'medium'</code>	Size/emphasis type of the snackbar

Outputs ■

Event	Type	Description
<code>dismissed</code>	<code>EventEmitter&lt;void&gt;</code>	Emitted when snackbar is dismissed
<code>actioned</code>	<code>EventEmitter&lt;string&gt;</code>	Emitted when action button is clicked

Methods ■

Method	Parameters	Return Type	Description
<code>show()</code>	<code>message: string, options?: SnackbarOptions</code>	<code>void</code>	Display a snackbar with the given message
<code>dismiss()</code>	<code>void</code>	<code>void</code>	Dismiss the currently displayed snackbar
<code>clear()</code>	<code>void</code>	<code>void</code>	Clear all displayed snackbars

CSS Custom Properties ■

Property	Description
<code>--snackbar-font-family</code>	Font family for snackbar text
<code>--snackbar-size-sm-font</code>	Font size for small snackbar
<code>--snackbar-size-md-font</code>	Font size for medium snackbar
<code>--snackbar-size-lg-font</code>	Font size for large snackbar
<code>--snackbar-success-color</code>	Color for success snackbar
<code>--snackbar-warning-color</code>	Color for warning snackbar
<code>--snackbar-error-color</code>	Color for error snackbar
<code>--snackbar-info-color</code>	Color for info snackbar
<code>--snackbar-font-weight-regular</code>	Regular font weight
<code>--snackbar-font-weight-semibold</code>	Semibold font weight
<code>--snackbar-subtitle-color</code>	Color for subtitle text
<code>--snackbar-title-color</code>	Color for title text
<code>--snackbar-radius-sm</code>	Border radius for snackbar
<code>--snackbar-shadowbox</code>	Box shadow for snackbar
<code>--snackbar-padding-small</code>	Padding for small snackbar
<code>--snackbar-padding-medium</code>	Padding for medium snackbar
<code>--snackbar-padding-large</code>	Padding for large snackbar
<code>--snackbar-padding-default</code>	Default padding for snackbar
<code>--snackbar-btn-font-sm</code>	Font size for small button
<code>--snackbar-padding-small-btn</code>	Padding for small button
<code>--snackbar-btn-font-md</code>	Font size for medium button

size for medium button --snackbar-padding-md-btn Padding for medium button  
 --snackbar-btn-font-large Font size for large button --snackbar-font-small Small font size  
 --snackbar-font-medium Medium font size --snackbar-font-large Large font size  
 --snackbar-font-large-title Font size for large title --snackbar-font-title Font size for title  
 --snackbar-font-subtitle Font size for subtitle --snackbar-padding-wrapper Horizontal padding for  
 snackbar container Accessibility ■ Keyboard accessible (focus, dismiss) Proper ARIA attributes  
 for screen readers Focus management and visible indicators High contrast and reduced motion  
 support Theming & Design Tokens ■ All colors, spacing, and effects are controlled via semantic  
 design tokens and CSS custom properties. Override these in your theme or component styles for  
 custom branding. Best Practices ■ Use snackbars for brief, non-blocking feedback Avoid placing  
 critical information solely in snackbars Provide clear, actionable messages Use actions for undo or  
 retry scenarios Test snackbar placement in responsive layouts Avoid excessive custom styling;  
 use built-in variants and types

```

import { Component, inject } from "@angular/core";
import { CommonModule } from "@angular/common";
import { SnackbarService, AavaButtonComponent } from "@aava/play-core";

@Component({
  standalone: true,
  imports: [CommonModule, AavaButtonComponent],
  template: `
    <div>
      <aava-button
        label="Show Snackbar"
        variant="primary"
        size="md"
        state="default"
        (click)="showSnackbar()"
      >
      </aava-button>
    </div>
  `,
})
export class SnackbarBasicUsageDemoComponent {
  private snackbar = inject(SnackbarService);

  showSnackbar() {
    this.snackbar.show("This is a snackbar message", "top-center");
  }
}

```

```

import { Component, inject } from "@angular/core";
import { CommonModule } from "@angular/common";
import { MatSnackBarService, MatSnackBarPosition } from "@aava/play-core";
import { AavaButtonComponent } from "@aava/play-core";

@Component({
  standalone: true,
  imports: [CommonModule, AavaButtonComponent],
  template: `
    <div>
      <aava-button
        *ngFor="let pos of positions"
        [label]="pos"
        variant="primary"
        size="md"
        state="default"
        (click)="showSnackBar(pos)"
      >
      </aava-button>
    </div>
  `,
})
export class SnackBarPositionsDemoComponent {
  private snackbar = inject(MatSnackBarService);

  positions: MatSnackBarPosition[] = [
    "top-left",
    "top-center",
    "top-right",
    "bottom-left",
    "bottom-center",
    "bottom-right",
    "center",
  ];

  showSnackBar(position: MatSnackBarPosition) {
    this.snackbar.show(`SnackBar at ${position}`, position);
  }
}

```

■ No code found